A method for the asynchronous transmission of information in accordance with ACID properties.

The invention relates to the field of transmitting information between suppliers and consumers via a string of communication channels. The invention consists more particularly of effecting such transmission asynchronously in accordance with ACID properties.

Asynchronous transfer mode communication systems are used more and more widely. Reliable communications are necessary in a great number of situations. Reliability can be assured by adding transactional mechanisms to the communication service.

An asynchronous communication service includes a manager and one or more communication channels.

A supplier (service client) sends information on a specific channel. The channel transfers the information to the consumer (also a service client) by invoking its interface. The channel acts as an intermediary between two clients and decouples the transmission of information at both ends. It must be able to deliver the information reliably and maintain the desynchronization of suppliers and consumers.

A prior art solution to the problem of transmitting data reliably is based on transactional systems. A transaction is initiated by a supplier of data and is then "propagated" to intermediate elements (communication channels) and eventually reaches a consumer. Once the data has been delivered, properties associated with the execution of the transaction are verified before confirming the modifications effected (i.e. the delivery of the data).

It must be possible for the execution of transactions to propagate via a communication channel. Asynchronous communication channels imply desynchronization of suppliers and consumers but transactions necessitate synchronization to enable the stability properties of the transactions to be monitored from end to end of execution.

A transaction must satisfy four properties grouped together under the acronym ACID: Atomicity, Coherence, Isolation and Durability. These four properties are closely interrelated. The objective of competition and restart control mechanisms executed by a transactional engine is to ensure compliance with them.

Atomicity guarantees that all the updates of a transaction are effected or none of them. Failure to comply with this property can change the set of data from a

coherent initial state to an incoherent state.

To comply with this property, any series of actions constituting a transaction is marked by a beginning and an end. The beginning of a transaction reports the start event of a transaction to the transactional engine. Two instructions are provided for marking the end of a transaction: "commit" enables the transaction to report to the transactional monitor that from the point of view of the transaction alone all of its actions have succeeded, and "abort" enables the transaction to report to the transactional monitor that one or more of its actions have failed and that the transaction does not wish to be validated (the data modified by the transaction must be returned to its preceding state). Depending on the event received by the transactional monitor at the end of the transaction, and on any overview of the system that it may have (interaction between transactions), the transactional monitor decides whether to validate the transaction, i.e. whether to finalize the modifications effected by it. If a system fault occurs before the end of a transaction the transaction is considered to be aborted. One mechanism for ensuring compliance with the atomicity property consists of retaining for each transaction in progress the preceding image of any data updated. If a transaction is aborted, it is possible to undo the transaction by applying all the preceding images of the transaction.

The coherence property concerns the semantic coherence of a set of data. This can be preserved in part by mechanisms that monitor integrity constraints and by maintaining the property [lacuna] of the transactions.

Isolation is indispensable in a multitasking environment to guarantee that each transaction sees a coherent state of the set of data. Isolation consists of guaranteeing that if the transaction is executed in parallel with other transactions accessing a common data set there is a serial execution of the same transactions that would produce the same changes on the data set accessed by the transactions. In this case, the isolation property is verified for that set of transactions.

Durability guarantees that the updates of a validated transaction are final. The only action that can permit the updates of a validated transaction to be undone is the execution of a compensation transaction. This property goes hand-in-hand with the atomicity property stipulating that the updates of a transaction form a coherent whole which is either aborted as a whole or validated permanently. A restart mechanism can be provided to recover information lost in the event of a memory fault or disk fault leading to the loss of some of the information from the database.

Reliable end-to-end asynchronous communication mechanisms are required.

It is difficult to combine the asynchronous aspect of the transfer with transactions which are essentially synchronous.

In a synchronous system, a transaction can be initiated and guarantee compliance with ACID properties from the information supplier to the consumer. In an asynchronous system the actions that guarantee these properties must interact with each other in a particular and reliable way, depending on the required quality of service.

The object of the invention is therefore to provide an asynchronous communication system having ACID properties.

The invention therefore provides a method of transmitting information asynchronously between a supplier and a consumer in accordance with ACID properties, the supplier and the consumer being connected by a chain of communication channels, characterized in that the information is transmitted by means of independent transactions set up:

- between the supplier and the first communication channel of the chain,
- between each of the communication channels of the chain, and
- between the last communication channel of the chain and the consumer.

The invention also provides a communication channel enabling asynchronous transmission of information between a supplier and a consumer in accordance with ACID properties, the channel having a set of clients which can be other communication channels and/or consumers, characterized in that it includes:

- means for storing the information contained in a transaction for which the channel is the target,
- means for finalizing the transaction, and
- means for initiating transactions containing the information with the clients if the transaction has succeeded.

The minimum number of transactions can be undone in the event of a problem because independent transactions are initiated for each link of the chain. This obviously achieves good performance in terms of the time to deliver the information to the consumer.

Also, the supplier initiates a transaction with only the first communication channel of the chain. The result of that transaction ("commit" or "abort") will therefore be obtained very quickly, and the supplier will not be blocked pending delivery of the information to the consumer. In other words, the objective of asynchronous transmission is achieved.

Delivery of the information to the consumer in compliance with ACID properties is ensured because all the intermediate transmissions are effected by means of transactions. In other words, it might be said that there is a "virtual transaction" between the supplier and the consumer.

The following detailed description of one embodiment of the invention explains the invention.

Figure 1 shows the general principle of a chain of transactions in accordance with the invention.

Figure 2 shows one example of a communication network using communication channels in accordance with the invention.

In figure 1, the suppliers, the asynchronous communication channel and the consumers are objects virtually connected into a unidirectional information propagation chain. As explained in more detail below, the chain can include a plurality of communication channels in cascade between the supplier and the consumers.

A transaction is initiated by a supplier 1 and is addressed to a communication channel 2. The transaction includes at least the information that the supplier wishes to transmit to the consumers 5.

The communication channel then stores the information in a reliable memory 4. The reliable memory can conventionally be a database which has an XA interface and the necessary restart mechanisms.

The transaction that was initiated by the supplier 1 is then finalized. If finalization succeeds (which conventionally corresponds to sending a "commit" message to the supplier), the information contained in the transaction is stored in queues 6. There is one queue for each client of the channel. A client of the channel can be a consumer, as in this example, or another communication channel. Thus in the figure 1 example there are two queues, each associated with one of the two consumers 5.

In one embodiment of the invention the information is stored in the queues before the transaction is finalized, but is not made available (for example by means of an availability identifier associated with each queue) until finalization has succeeded.

Note that more than one supplier can invoke the same communication channel 2 in parallel. This is made technically possible by using more than one thread.

Similarly, there is more than one thread for each consumer virtually connected to the channel. The role of the threads is to consume the information stored in the queues 6 in order to initiate transactions with the clients of the communication channel. Those clients can be consumers 5 or, obviously, other communication channels.

The transactions contain the information stored in the queues and previously received from the supplier 1. Thus the information is transmitted along the chain.

The clients (here the consumers 5) then finalize the transactions. If the transactions succeed, the communication channel receives in return a message ("commit") advising it of this. In this case, the information stored in the queue 6 corresponding to the client that transmitted the "commit" is withdrawn.

Note that in this case the reliable memory is never accessed in read mode. Because the reliable memory is typically a database, it is clear that any access to it represents a penalty in terms of execution speed. The above method therefore minimizes access to the reliable memory by using much faster queues and consequently increases the speed at which information is transmitted from the supplier to the consumers.

In one embodiment of the invention each software element (supplier, communication channel, consumers) provides a standard recovery interface for restarting in the event of a fault. That interface enables the action performed by a transaction that has failed to be undone. Similarly, the reliable memory provides an interface mainly enabling information associated with a transaction to be withdrawn from storage. Accordingly, if one of the transactions in the chain fails, all of the actions undertaken on the objects implicated by the transaction in question are undone.

Once the actions have been undone because of the failure of a transaction, the transaction can be restarted by reading the information stored in the reliable memory.

As stated above, if a transaction fails, it can be undone and restarted without calling into question the other transactions previously effected in the chain.

If the state of a channel, i.e. the content of the queues and the knowledge of its virtual interconnections, is lost, the reliable memories enable it to be restored.

The above solution provides an end-to-end data recovery mechanism working automatically with an external transactional system. Compared to other solutions, the mechanism is generic.

The foregoing description considers only one communication channel, but the mechanism is symmetrical and works in the same way in any chain of channels. In other words, there can be any number of communication channels between the supplier and the consumer or consumers. As previously indicated, in this situation a communication channel can be connected as a client of another communication channel.

Figure 2 shows one example of a network of communication channels. A supplier F_1 is connected to a first communication channel. A supplier F_2 is connected to a second communication channel. These two communication channels have a third communication channel as a client. One or more consumers C_1 are connected as clients of the first communication channel and one or more consumers C_2 are connected as clients of the second communication channel.

The above kind of network topology is highly beneficial in practice. It enables consumers C_1 to receive only information from suppliers F_1 and consumers C_2 to receive information from suppliers F_1 and F_2. Any software entity can therefore choose to receive information from more than one supplier by choosing the communication channel to which it must connect as a client.